Amendments to the Specification:

Please replace the paragraphs at page 1, line 4, through page 2, line 15, with the following amended paragraphs:

The increasing demand for memory bandwidth in recent high-performance VLSI processors is not satisfied by existing memory technology. In "Fault-Tolerant Interleaved Memory Systems with Two-Level Redundancy", by Lu et al in IEEE Transactions on Computers, Vol. 46, No. 9, September 1997, several memory banks or modules of a main memory are accessed by processors in an interleaved fashion, in order to achieve a memory with high bandwidth. However, if a plurality of memory modules are used, some of these modules might be faulty. To cope with these faulty modules Lu et al ~~purpose~~ proposes to use a memory containing a plurality of modules as well as to provide some spare modules in said memory. These spare modules may belong to the same bank or may constitute global spare modules. If a faulty module occurs, the memory management may initiate a replacement of this faulty module by one of the spare modules. A module map table is provided for selecting a spare module to replace a faulty one and a bank map table is provided for selecting a spare bank to replace a faulty bank. Nonetheless, the teachings of Lu et al relate to the communication between processors and main memory having redundant memory modules without any caches between the processors and the main memory, which does not appear to be beneficial in terms of latency and bandwidth.

In contrast to the above, the extensive use of on-chip cache memories have became essential to sustain the memory bandwidth demand of the CPU. The advances in semiconductor technology and continuous down scaling of feature size creates extra-space for additional functionality on single chip. The most popular way to make use of this extra space is integrating a cache of bigger size so that a microprocessor is able to gain higher performance. However, an increase in the circuit density is closely coupled with an increase in probability of defects. Caches constitute a redundant structure which is employed to enhance the performance of the CPU. One method to tolerate faults in the cache is providing spare cache blocks. The defective block is switched to ~~spare~~ a spare

block by a reconfiguration mechanism, or by providing small fully associative cache to dynamically replace the faulty block.

However, since the provision of caches with spare or redundant memory modules is expensive, new techniques being able to sooth the degradation of cache performance without spare cache blocks are needed. Therefore, instead of using explicit spare blocks, the physical or logical neighborhood blocks play the role of spare block. Dong-Hyun et al, "Re-evaluation and Comparison of Fault Tolerant Cache Schemes", University of Wisconsin Madison ECE Dept. 753 Course Project, 2002, as well as Shirvani et al, "PADded Cache: A New Fault-Tolerance Technique for Cache Memories", Proc. 17.sup.th IEEE VLSI Test Symposium, 1999 describe a Programmable Address Decoder PAD for a cache. A PAD is a decoder which has programmable mapping function. As ~~mention~~ mentioned before, caches have an intrinsic redundancy since the purpose of caches is to improve performance. Many processing architectures can work without any cache but at the cost of degraded performance. Therefore, introducing additional redundancy, like spare memory blocks, is inefficient.

Please replace the paragraphs at page 2, line 25, through page 3, line 16, with the following amended paragraphs:

There are three different ways that the mapping can be performed. In a Direct Mapped Cache, which is the simplest way to allocate the cache to the system memory, it is determined how many cache lines are present and the system memory is divided into the same number of ~~portion.~~ portions. Then each portion is used by one cache line. The Fully Associative Cache makes it possible to design a cache, such that any line can store the contents of any memory location, instead of hard-allocating cache lines to particular memory locations. The third cache mapping scheme is the N-Way Set Associative Cache. This scheme constitutes a compromise between the direct mapped and fully associative designs. The cache is divided into sets, where each set contains N cache lines, i.e. N ways. Then, each memory address is assigned to a set, and can be cached in any one of those N locations within the set that it is assigned to. In other words, within each set the cache is associative. Accordingly, there are "N" possible places that a given memory

location may be in the cache. The mapping is usually integrated in a tag RAM address decoder, which constitutes the area in an L2 cache that identifies which data from main memory is currently stored in each cache line. The values stored in the tag RAM determine whether a cache lookup results in a hit or a miss.

For instance, each way of a 4-way associative cache can have separate PADS. Therefore, cache addresses for faulty blocks are remapped to correct blocks within said way. All addresses are still cacheable, but conflict misses are increased. For direct-mapped caches, since at least one address bit information is lost as a consequence of remapping, at least one bit is augmented to the tag bits in order to be able to distinguish those addresses that may be mapped to the same block. The cache remapping is performed on a per-block basis, wherein a faulty block is mapped to a "healthy" one, which address differs from the address of the faulty block by merely one bit. Usually, for set-associative caches a separate memory array is provided for each way, so that a decoder can be associated to each array. Accordingly, the remapping is performed merely in one array or way and will not affect the mapping of the other arrays.

Please replace the paragraph beginning on page 4, line 19, with the following amended paragraph:

The invention is based on the idea to improve the yield of a cache by exploiting the redundancy within a cache. In a cache entire memory modules can be interchanged such that faulty modules are not used anymore. The available working modules may be reshuffled such that each memory bank comprises at least one way. Preferably, the reshuffling is ~~preformed~~ performed such that the faulty modules are distributed evenly over the memory banks, whereby the highest (guaranteed) associativity and therefore performance is achieved.

Please replace the paragraph beginning on page 5, line 3, with the following amended paragraph:

Fig. 1 shows an overall system architecture, where the principles of the invention may be implemented. This architecture may constitute a heterogeneous shared memory system containing 8 processing units TM, one MIPS processor mips, a DDR double data rate DRAM controller DDR_ctrl, and a 12 Mb shared L2 cache L2_bank. The caches L2_bank, the processing units TM, the controller DDR_ctrl as well as the processor mips are connected to each other via a coherent interconnection network CIN, which also comprises a cache controller CC. A network interface controller NIC is provided to handle the high-speed communication with other ~~part~~ parts of the overall processing system. The processing unit is preferably a Trimedia ~~processors~~ processor running at 400 MHz.

Please replace the paragraphs at page 6, line 10, through page 7, line 2, with the following amended paragraphs:

Fig. 3 depicts the select signal circuitry for the L2 cache according to a first embodiment. In Fig. 3 a tag RAM unit TagRAM, a remapping means RM, two registers R and a plurality of comparators C are shown in Fig. 3. The pipeline registers R receive the address Ad as input signal, while one register R provides a tag reference signal tag_ref as output, and the other register provides a bank selection signal bank_select and a word selection signal word_select as output ~~signal~~. The tag RAM unit TagRAM receives the address Ad as input ~~signal~~ and outputs the signals tag0, . . . , tag5. These signals and the tag reference signal tag_ref are input signals of the comparators C, respectively, which output the hit signals hit0, . . . , hit5. These hit signals, the bank select signal bank_select and the word selection signal word_select are input to the remapping means RM, where the remapping operation is performed and the input signals are mapped to hit'0, . . . hit'5, bank'select, and word'select representing the new remapped position of the cache module. The function and operation of the Tag RAM unit is well known in the art and will therefore not be described in detail.

The signal for selecting a bank bank_select is used to select one of the eight banks bank0 to bank7. The hit signal hit0, . . . ,hit5 identifies a way in the bank selected by the bank select signal bank_select. The word selecting signal word_select is used to access a word in the (32 k 64) way. The remapping means RM is placed after the hit and bank select signals, i.e. in series to the Tag RAM unit TagRAM. The remapping means RM preferably implements a module permutation function which results in bank_select' and hit0' . . . hit5' selecting remapped modules. The permutation function is a process of generating an alternative arrangements of given data and can for example be implemented with a register file of 48 (6*8) registers of 9 bits each (with 6 ways selects plus 3 (encoded) bank selects). The permutation is ~~preformed~~ performed without reducing the number of inputs. Additionally, the word_select signals may also be remapped. This can be desirable if a way-bank combination itself consists of multiple memory modules. Alternatively, the remapping can be performed on the basis of a reduction mapping, i.e. less output symbols ~~then~~ than input symbols.

Please replace the paragraphs at page 7, lines 5-24, with the following amended paragraphs:

Fig. 4 shows an alternative implementation for remapping modules according to a second embodiment. The selection circuitry is based on the selection circuitry according to the first embodiment, however the remapping means RM is replaced by a Map RAM unit MapRAM. The MapRAM may be implemented as an ordinary RAM with its address being a bank number, and outputting 9 bits for each of the 6 ways. These bits consist of the remapped way and the remapped bank (together addressing a remapped module). While according to the first embodiment the remapping unit RM is implemented in series to the Tag RAM unit, the Map RAM unit is implemented in parallel to the Tag RAM unit. The Map RAM unit receives the address Ad as input ~~signal~~ and outputs a mapping signal map0, . . . map5. Accordingly, the remapping for each way of the addressed bank is looked up in parallel with the tag lookup, i.e. the Tag RAM unit. The mapping signals map0, . . . , map5 as well as the hit signals hit0, . . . hit5 from the comparators C constitute input signals into six AND-Gates G, respectively. The outputs

of these gates G are fed to an OR-Gate H. The output of the gate H constitutes the way select signal way_select and the bank select signal bank_select. Furthermore, the address is inputted to a register R, which outputs the word select. The hitting way then selects which of the 6 pre-selected remappings map0, . . . , map5 is to be performed.

The registers R in the second embodiment ~~is~~ are not essential. The AND gates and OR gates occur 6*9 times and 9 times, respectively. However, this function may also be implemented for example using a multiplexer.

Please replace the paragraph beginning on page 8, line 1, with the following amended paragraph:

According to a third embodiment, which is based on the first embodiment, i.e. arranging the Tag RAM unit and the remapping means RM in series, the remapping may also be performed at a finer granularity, e.g. on a block/line level instead of module level according to the first and second embodiment. This results in an additional circuit modifying the addresses indexing the tag RAM. A preferred mapping is performed by remapping certain addresses to the same index, e.g. by mapping block0 and block1 to the same index/block by ignoring the least significant address bit. If multiple addresses map to the same index, the tag RAM is augmented with the missing address information. A widened tag comparison can be used to resolve the ambiguity. This has the advantage, that it can easily be integrated with the address decoder. If the ambiguity problem (with wider tags and tag comparisons) is to be avoided, then the address mapping has to be a permutation which in turn is impractical at a block level. Moreover, address remapping becomes less attractive in cases of a hard RAM macro with ~~integrated~~ an integrated address decoder.

Please replace the paragraphs at page 8, lines 19-31, with the following amended paragraphs:

In Fig. 6 a look-up table is shown, which represents a look up table of a combined map RAM and the tag RAM according to the fourth embodiment. Additionally, ~~to~~

compared with the look up table according to Fig. 5, the table according to Fig. 6 comprises an extra column, i.e. the map column, for each way. This column contains the remapping information, e.g. the block at (index0,bank0,way2) gets remapped to (index0,bank2,way5), since the values 2 and 5 are written into the corresponding map field. In order to realise a cheap and fast implementation, a block is preferably not remapped to a different index, but merely to a new way and bank. However, the remapping may alternatively also be performed over several indexes.

~~For~~ The remapping means may comprise a programmable permutation unit which indicates the module-to-module mapping, a circuit using this permutation to reshuffle modules by recomputing module selects, and/or a mechanism to permanently invalidate the tag rams associated with a faulty module (after remapping).